IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

U.S. PATENT APPLICATION NUMBER 10/762,866
FILING DATE ..... 22 January 2004
INVENTOR ..... Kevin J. Turpin
ASSIGNEE ..... Symantec Corporation
GROUP ART UNIT 2167
EXAMINER ..... Robert M. Timblin
ATTORNEY'S DOCKET NO. 4001-0121
TITLE ..... METHOD AND APPARATUS FOR LOCALIZED PROTECTED IMAGING OF A FILE SYSTEM

APPEAL BRIEF

To: MS Appeal Brief – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

From: Jonathan R. Lee
ADVANTEDGE LAW GROUP, LLC
922 W. Baxter Drive
South Jordan, UT 84095
Telephone: (801) 285-5175
Facsimile: (866) 259-6138

Dear Sir:

In accordance with 37 C.F.R. § 41.37(a), Appellant respectfully submits this Appeal Brief in furtherance of the Notice of Appeal filed in the above-identified application on 6 April 2010, which appeals the Non-Final Office Action dated 1 March 2010. In compliance with 37 C.F.R. § 41.37(a)(1), Appellant submits one (1) copy of this Appeal Brief.

I. REAL PARTY IN INTEREST

Symantec Corporation is the real party in interest in the present application. An assignment of all rights in the present application to Symantec Corporation was recorded by the U.S. Patent and Trademark Office on 6 September 2007 at Reel 019781, Frame 0651.

II. RELATED APPEALS, INTERFERENCES, AND JUDICIAL PROCEEDINGS

A notice of Appeal and a request for Pre-Appeal Conference were previously filed in this case on 12 March 2008. A Notice of Panel Decision from Pre-Appeal Brief review was mailed on 24 April 2008. The Panel reopened prosecution. Appellant is not aware of any other appeals, interferences, or judicial proceedings that will directly affect, be directly affected by, or have a bearing on the Board's decision in this appeal.

III. JURISDICTIONAL STATEMENT

In accordance with 37 C.F.R. § 41.37(a), Appellant respectfully submits this Appeal Brief in furtherance of the Notice of Appeal filed in the above-identified application on 6 April 2010, which appeals the Non-Final Office Action dated 1 March 2010. The fees required under 37 C.F.R. § 41.20(b)(2) are provided in the accompanying TRANSMITTAL OF APPEAL BRIEF.

IV. TABLE OF CONTENTS

V. TABLE OF AUTHORITIES

VI. STATUS OF AMENDMENTS

No amendments to the claims have been made subsequent to the Final Office Action dated 1 March 2010, which Action is the subject of this Appeal. A copy of the pending claims is attached to this Brief in the Appendix.

VII. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1, 11, 21, 31, 41, 42, and 43 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,615,365 to Jenevein et al. ("Jenevein") in view of U.S. Patent No. 6,026,016 to Gafken ("Gafken"). Appellant respectfully requests that these grounds of rejection be reviewed in the instant Appeal.

VIII. STATEMENT OF FACTS

In the rejection of independent claims 1, 11, 21, 31, 41, 42, and 43, the Examiner recognizes that "Jenevein does not appear to teach protection by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file." Final Office Action dated March 1, 2010, page 3.

In an attempt to remedy the conceded deficiencies of Jenevein, the Examiner asserts that Gafken "teaches initiating a process at system startup (col. 3 lines 24-31 and col. 12 lines 8-11) that opens (fig. 5, e.g. numeral 525) the locally-stored image file to block subsequent processes from accessing the locally-stored image file (col. 3, lines 29-31, col. 12 lines 8-11) for proving [*sic.*] a locking mechanism that operates during start-up to protect critical information stored in blocks from undesired alterations (Gafken, col. 10, lines 30-33)." Final Office Action dated

March 1, 2010, page 3.

Gafken is directed to a "hardware block locking" system that uses a block-locking circuit 140 to prevent "viruses and other destructive sources of memory corruption" from altering data (such as a system's BIOS) stored in specific blocks of memory within a memory array 130. Col. 14, lines 27-39; col. 4, lines 27-34; Fig. 1. According to Gafken, block locking circuit 140 includes a lock bit array 315 that contains a plurality of sets of protection bits "arranged in segments 0-N to correspond respectively to blocks 0-N of the memory array 130." Col. 6, lines 4-59. Gafken also explains that each set of protection bits within lock bit array 315 "stores a value indicating the protection status ["locked, unlocked, or locked-down"] of the corresponding block of the memory array [that is to be protected]." *Id.* Thus, by changing a protection bit set's value to "locked" or "locked-down," the system in Gafken may prevent a corresponding block of memory within memory array 130 (which, according to Gafken, is often "used to store the start-up and/or basic input/output system (BIOS) routines that are executed when a computer system is reset or turned on") "from being accessed for program or erase operations." *Id.*; col. 1, lines 19-22.

IX. ARGUMENT

In the Action, claims 1, 11, 21, 31, 41, 42, and 43 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,615,365 to Jenevein et al. ("Jenevein") in view of U.S. Patent No. 6,026,016 to Gafken ("Gafken"). 35 U.S.C. § 103(a) recites, in part:

> A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains.

To establish prima facie obviousness of a claimed invention, "[a]ll words in a claim must be considered in judging the patentability of that claim against the prior art." *In re Wilson*, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970). Appellant respectfully submits that the cited references, even if combined, do not establish a *prima facie* case of obviousness because they do not show, teach, or suggest all claimed features.

A. The Cited References Fail to Disclose, Teach, or Suggest the Protection Feature Recited in Independent Claims 1, 11, 21, 31, 41, 42, and 43

As will be explained in greater detail below, the cited references fail to teach the protection feature of claims 1, 11, 21, 31, 41, 42, and 43. In the Final Office

Action, the Examiner acknowledges that Jenevein does not teach of "protecting [a] locally-stored image file from accidental user deletion of modification by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file," as recited in independent claims 1, 11, 12, 31, 41, 42, and 43. Final Office Action, mailed March 1, 2010, page 3. However, in an attempt to remedy these conceded deficiencies, the Examiner alleges that:

> "Gafken ... teaches initiating a process at system startup (col. 3 lines 24-31 and col. 12 lines 8-11) that opens (fig. 5, e.g. numeral 525) the locally-stored image file to block subsequent processes from accessing the locally-stored image file (col. 3, lines 29-31, col. 12 lines 8-11) for proving [*sic*.] a locking mechanism that operates during start-up to protect critical information stored in blocks from undesired alterations (Gafken, col. 10, lines 30-33)."

Id.

Gafken does not, however, teach of protecting a locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file. Instead, and as will be described in greater detail below, Gafken merely describes using a

hardware circuit to regulate access to specific memory blocks within a nonvolatile memory that contains a system's basic input/output system (BIOS).

Gafken describes a "hardware block locking" system that uses a block-locking circuit 140 to prevent "viruses and other destructive sources of memory corruption" from altering data (such as a system's BIOS) stored in specific blocks of memory within a memory array 130. Col. 14, lines 27-39; col. 4, lines 27-34; Fig. 1. According to Gafken, block locking circuit 140 includes a lock bit array 315 that contains a plurality of sets of protection bits "arranged in segments 0-N to correspond respectively to blocks 0-N of the memory array 130." Col. 6, lines 4-59. Gafken also explains that each set of protection bits within lock bit array 315 "stores a value indicating the protection status ["locked, unlocked, or locked-down"] of the corresponding block of the memory array [that is to be protected]." *Id.* Thus, by changing a protection bit set's value to "locked" or "locked-down," the system in Gafken may prevent a corresponding block of memory within memory array 130 (which, according to Gafken, is often "used to store the start-up and/or basic input/output system (BIOS) routines that are executed when a computer system is reset or turned on") "from being accessed for program or erase operations." *Id.*; col. 1, lines 19-22.

In summary, in order to securely update a system's BIOS stored within memory array 130, the system disclosed in Gafken must: 1) unlock access to the blocks of memory within memory array 130 that contain the system's BIOS by changing the value of corresponding protection bits within lock bit array 315 from "locked" or "locked down" to "unlocked," 2) update/flash the system's BIOS by copying a new BIOS image ("obtained from a mass storage device ... or over a network or from the World Wide Web") to the blocks of memory within memory array 130 that contain the system's BIOS, and then 3) re-lock the blocks of memory within memory array 130 that contain the updated BIOS by changing the value of the corresponding protection bits within lock bit array 315 from "unlocked" to "locked down" or "locked." See, e.g., col. 12, lines 38-48 and col. 13, line 26 to col. 14, line 10.

As is clear from the above summary, Gafken clearly fails to teach of protecting memory blocks within a memory array 130 that contain a system's current BIOS by opening these blocks during system startup, let alone protecting a locally stored image file by initiating a process at system startup that opens the locally stored image file, as recited in the independent claims of the instant application. While col. 12, lines 8-11 of Gafken teaches that, during a BIOS update process, boot code may maintain exclusive control of system 100 in order

to ensure that "viruses, system software and/or other processor commands cannot alter the information stored in the memory array 130 unless the boot code 330 specifically provides for the changes," Gafken fails to teach or suggest that this boot code protects the memory blocks within memory array 130 that contain the system's BIOS by opening these memory blocks. Instead, Gafken merely teaches of using a hardware circuit (block locking circuit 140) to protect access to these memory blocks, as detailed above.

In the most-recent Final Office Action, the Examiner appears to argue that Gafken protects blocks of memory within memory array 130 that contain a system's BIOS by initiating a process at system startup that opens a separate "code image" that is "to be used to update code [such as a system's BIOS] previously stored in the memory array 130." Final Office Action, mailed March 1, 2010, page 3. The Examiner's argument fails for at least the following reasons: 1) Gafken teaches that the blocks of memory within memory array 130 that contain the system's BIOS are protected by activating specific protection bits within lock bit array 315 of block locking circuit 140, *not* by opening a "code image" that contains an updated BIOS image and 2) even if (for the sake of argument) Gafken taught of initiating a process during system startup that opens an image file (such as the "code image" disclosed in Gafken), the independent claims of the instant

application recite "protecting [a] *locally-stored image file* … by initiating a process at system startup that *opens the locally stored image file*," not protecting *blocks of memory in a nonvolatile memory array* by opening a separate image file. The Examiner appears to be using the claims of the instant application as a blueprint for attempting to read features into Gafken that are not taught or suggested anywhere in Gafken.

Moreover, the "code image" disclosed in Gafken and cited by the Examiner is clearly not analogous to the locally stored image file (which, according to the claims of the instant application, contains a backup of a file system) recited in the claims of the instant application. For example, Gafken clearly states that this code image is "used to update code [such as a system's BIOS] previously stored in the memory array 130." Col. 12, line 38-44 and Fig. 1. In other words, the code image described by Gafken merely contains code (such as a new BIOS image) to be used to update the blocks of memory within memory array 130. This is clearly different from a "locally-stored image file" that contains "a plurality of files of [a] file system" (i.e., a backup of a file system) and is "located within the same partition as the file system being backed up," as recited in the claims of the instant application.

For at least the above reasons, Gafken fails to teach or suggest "protecting the locally-stored image file from accidental user deletion or modification by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file," as recited in claims 1, 11, 21, 31, 41, 42, and 43. Accordingly, Appellant submits that claims 1, 11, 21, 31, 41, 42, and 43 distinguish over the proposed combination of Jenevein and Gafken and are in condition for allowance.

X. APPENDIX

A. Claim Section

The following listing of claims is a clean copy of all claims pending the application.

1. (Rejected) A method for backing up a file system in a partition comprising a plurality of allocation units, the method comprising:

creating a locally-stored image file by copying each allocation unit occupied by a plurality of files of the file system to the locally-stored image file, wherein the locally-stored image file is located within the same partition as the file system being backed up;

adding a directory map to the locally-stored image file that associates copied allocation units in the locally-stored image file with names of corresponding files from the file system; and

subsequent to creating the locally-stored image file, protecting the locally-stored image file from accidental user deletion or modification by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file.

2. (Rejected) The method of claim 1, wherein copying comprises compressing at least a subset of the allocation units.

3. (Rejected) The method of claim 1, wherein copying comprises:

maintaining a record of a pre-imaging state of the file system; and

copying only allocation units occupied by files included within the pre-imaging state of the file system.

4. (Rejected) The method of claim 1, wherein adding comprises grouping within the locally-stored image file the copied allocation units for individual files of the file system.

5. (Rejected) The method of claim 1, wherein copying comprises storing within the locally-stored image file one or more attributes related to each file, wherein the attributes comprise at least one of ownership attributes, access-control attributes, timestamp attributes, archival attributes, indexing attributes, encryption attributes, and compression attributes.

6. (Rejected) The method of claim 1, further comprising marking a beginning point of the locally-stored image file to assist in locating the locally-stored image file in the event of directory area corruption.

7. (Rejected) The method of claim 6, wherein marking comprises storing a unique beginning-of-image marker at an initial allocation unit occupied by the locally-stored image file.

8. (Rejected) The method of claim 6, wherein marking comprises storing, at a predetermined area of the partition, a location of an initial allocation unit occupied by the locally-stored image file.

9. (Cancelled)

10. (Rejected) The method of claim 1, wherein protecting the locally-stored image file further comprises providing a filter driver that intercepts and denies requests to access the locally-stored image file.

11. (Rejected) A method for restoring a file system to a partition comprising a plurality of allocation units, the method comprising:

accessing a locally-stored image file located within the partition to which the file system is to be restored, the locally-stored image file comprising a directory map and file data for a plurality of files;

initializing at least a subset of the allocation units of the partition not occupied by the locally-stored image file including one or more allocation units used for a directory area of the partition;

extracting the file data from the locally-stored image file into the initialized allocation units without disturbing the locally-stored image file;

creating a new directory area for the partition using the directory map; and

protecting the locally-stored image file from accidental user deletion or modification subsequent to creation of the locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file.

12. (Rejected) The method of claim 11, wherein the directory map associates names for the plurality of files with corresponding portions of the file data, and wherein creating comprises generating a new directory area for the partition that associates the file names with the extracted file data.

13. (Rejected) The method of claim 11, wherein creating comprises adding an indication of the locally-stored image file to the new directory area.

14. (Rejected) The method of claim 11, wherein extracting comprises decompressing at least a subset of the file data.

15. (Rejected) The method of claim 11, wherein the directory map indicates at least one attribute for a file, and wherein creating comprises setting the at least one attribute for the file in the directory area, wherein the at least one attribute comprises at least one of an ownership attribute, an access control attribute, a timestamp attribute, an archival attribute, an indexing attribute, an encryption attribute, and a compression attribute.

16. (Rejected) The method of claim 11, wherein accessing comprises searching for an allocation unit containing a unique beginning-of-image marker for the locally-stored image file.

17. (Rejected) The method of claim 11, wherein accessing comprises reading from a predetermined area of the partition a location of an initial allocation unit of the locally-stored image file.

18. (Rejected) The method of claim 11, further comprising defragmenting the locally-stored image file within the partition prior to extracting the file data.

19. (Cancelled)

20. (Rejected) The method of claim 11, wherein protecting the locally-stored image file further comprises providing a filter driver that intercepts and denies requests to access the locally-stored image file.

21. (Rejected) An apparatus for backing up a file system in a partition comprising a plurality of allocation units, the apparatus comprising:

a processor;

a local imager programmed to create a locally-stored image file by copying each allocation unit occupied by a plurality of files of the file system to the locally-stored image file, wherein the locally-stored image file is located within the same partition as the file system being backed up, and wherein the local imager is configured to add a directory map to the locally-stored image file that associates copied allocation units in the locally-stored image file with names of corresponding files from the file system; and

a protection component programmed to protect the locally-stored image file from accidental user deletion or modification subsequent to creation of the locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file.

22. (Rejected) The apparatus of claim 21, wherein the local imager is configured to compress at least a subset of the allocation units copied to the locally-stored image file.

23. (Rejected) The apparatus of claim 21, wherein the local imager is configured to maintain a record of a pre-imaging state of the file system and to copy only allocation units occupied by files included within the pre-imaging state of the file system.

24. (Rejected) The apparatus of claim 21, wherein the local imager is configured to group within the locally-stored image file the copied allocation units for individual files of the file system.

25. (Rejected) The apparatus of claim 21, wherein the local imager is configured to store within the locally-stored image file one or more attributes relating to at least one file of the file system, wherein the file attributes comprise at least one of ownership attributes, access-control attributes, timestamp attributes, archival attributes, indexing attributes, encryption attributes, and compression attributes.

26. (Rejected) The apparatus of claim 21, wherein the local imager is configured to mark a beginning point of the locally-stored image file to assist in locating the locally-stored image file in the event of directory area corruption.

27. (Rejected) The apparatus of claim 26, wherein the local imager is configured to mark the beginning point by storing a unique beginning-of-image marker at an initial allocation unit occupied by the locally-stored image file.

28. (Rejected) The apparatus of claim 26, wherein the local imager is configured to mark the beginning point by storing, at a predetermined area of the partition, a location of an initial allocation unit occupied by the locally-stored image file.

29. (Cancelled)

30. (Rejected) The apparatus of claim 21, wherein the protection component further comprises a filter driver that intercepts and denies requests to access the locally-stored image file.

31. (Rejected) An apparatus for restoring a file system to a partition comprising a plurality of allocation units, the apparatus comprising:

a processor;

an image locater to find a locally-stored image file located within the partition to which the file system is to be restored, the locally-stored image file comprising a directory map and file data for a plurality of files;

a media formatter to initialize at least a subset of the allocation units of the partition not occupied by the locally-stored image file including one or more allocation units used for a directory area of the partition;

a data extractor to extract the file data from the locally-stored image file into the initialized allocation units without disturbing the locally-stored image file;

a directory area builder to build a new directory area for the partition using the directory map; and

a protection component programmed to protect the locally-stored image file from accidental user deletion or modification subsequent to creation of the locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file.

32. (Rejected) The apparatus of claim 31, wherein the directory map associates names for the plurality of files with corresponding portions of the file data, and wherein the directory area builder is configured to generate a new directory area for the partition that associates the file names with the extracted file data.

33. (Rejected) The apparatus of claim 31, wherein the directory area builder is configured to add an indication of the locally-stored image file to the new directory area.

34. (Rejected) The apparatus of claim 31, wherein the data extractor is configured to decompress at least a subset of the file data.

35. (Rejected) The apparatus of claim 31, wherein the directory map indicates at least one attribute for a file, wherein the directory area builder is configured to set the at least one attribute of the file in the directory area, and wherein the at least one attribute comprises at least one of an ownership attribute, an access control attribute, a timestamp attribute, an archival attribute, an indexing attribute, an encryption attribute, and a compression attribute.

36. (Rejected) The apparatus of claim 31, wherein the image locater is configured to search for an allocation unit containing a unique beginning-of-image marker for the locally-stored image file.

37. (Rejected) The apparatus of claim 31, wherein the image locater is configured to read from a predetermined area of the partition a location of at least a first allocation unit of the locally-stored image file.

38. (Rejected) The apparatus of claim 31, further comprising an image defragmenter to defragment the locally-stored image file within the partition before the data extractor extracts the file data.

39. (Cancelled)

40. (Rejected) The apparatus of claim 31, wherein the protection component further comprises a filter driver that intercepts and denies requests to access the locally-stored image file.

41. (Rejected) A method for localized backup and restoration of a file system in a partition comprising a plurality of allocation units, the method comprising:

creating a locally-stored image file by copying each allocation unit occupied by a plurality of files of the file system to the locally-stored image file, wherein the locally-stored image file is located within the same partition as the file system being backed up;

adding a directory map to the locally-stored image file that associates copied allocation units in the locally-stored image file with names of corresponding files from the file system;

locating the locally-stored image file within the partition;

initializing at least a subset of the allocation units of the partition not occupied by the locally-stored image file including one or more allocation units used for a directory area of the partition;

extracting file data from the locally-stored image file into the initialized allocation units without disturbing the locally-stored image file;

creating a new directory area for the partition using the directory map; and

subsequent to creating the locally-stored image file, protecting the locally-stored image file from accidental user deletion or modification by initiating a

process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file.

42. (Rejected) A computer-readable storage medium comprising program code for backing up a file system in a partition comprising a plurality of allocation units, the computer-readable storage medium comprising:

program code for creating a locally-stored image file by copying each allocation unit occupied by a plurality of files of the file system to a locally-stored image file, wherein the locally-stored image file is located within the same partition as the file system being backed up;

program code for adding a directory map to the locally-stored image file that associates copied allocation units in the locally-stored image file with names of corresponding files from the file system; and

program code for protecting the locally-stored image file from accidental user deletion or modification subsequent to creation of the locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file.

43. (Rejected) A computer-readable storage medium comprising program code for restoring a file system to a partition comprising a plurality of allocation units, the computer-readable storage medium comprising:

program code to access a locally-stored image file located within the partition to which the file system is to be restored, the locally-stored image file comprising a directory map and file data for a plurality of files;

program code to initialize at least a subset of the allocation units of the partition not occupied by the locally-stored image file including one or more allocation units used for a directory area of the partition;

program code to extract the file data from the locally-stored image file into the initialized allocation units without disturbing the locally-stored image file;

program code to create a new directory area for the partition using the directory map; and

program code to protect the locally-stored image file from accidental user deletion or modification subsequent to creation of the locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file.

B. Claim Support and Drawing Analysis Section

1. (Rejected) A method for backing up a file system in a partition comprising a plurality of allocation units, the method comprising:

creating a locally-stored image file by copying each allocation unit occupied by a plurality of files of the file system to the locally-stored image file **{page 5, paragraph 17; page 6, paragraph 18; page 11, paragraph 42; page 12, paragraphs 46 and 49; page 20, paragraph 77; Fig. 3, 204, 302, 306; Fig. 9, 902}**, wherein the locally-stored image file is located within the same partition as the file system being backed up **{page 5, paragraph 17; page 11, paragraphs 42, 44, and 45; page 12, paragraph 49; page 20, paragraph 77; Fig. 2, 102, 104, 204; Fig. 9, 902}**;

adding a directory map to the locally-stored image file that associates copied allocation units in the locally-stored image file with names of corresponding files from the file system **{page 5, paragraph 17; page 14, paragraph 54; page 18, paragraph 70; page 19, paragraph 78; page 21, paragraph 83; Fig. 3, 312; Fig. 9, 904}**; and

subsequent to creating the locally-stored image file, protecting the locally-stored image file from accidental user deletion or modification by initiating a process at system startup that opens the locally-stored image file to block

subsequent processes from accessing the locally-stored image file **{page 6, paragraphs 18 and 22; page 11, paragraph 45; page 15, paragraph 60; page 16, paragraph 62; page 20, paragraph 79; page 21, paragraph 83; Fig. 9, 906}**.

11. (Rejected) A method for restoring a file system to a partition comprising a plurality of allocation units, the method comprising:

accessing a locally-stored image file located within the partition to which the file system is to be restored **{page 6, paragraph 20; page 17, paragraph 66; page 21, paragraph 81; Fig. 9, 908}**, the locally-stored image file comprising a directory map and file data for a plurality of files **{page 5, paragraph 17; page 14, paragraph 54; page 18, paragraph 70; page 19, paragraph 78; page 21, paragraph 83; Fig. 3, 312; Fig. 9, 904}**;

initializing at least a subset of the allocation units of the partition not occupied by the locally-stored image file including one or more allocation units used for a directory area of the partition **{page 6, paragraph 21; page 17, paragraph 68; Fig. 9, 910}**;

extracting the file data from the locally-stored image file into the initialized allocation units without disturbing the locally-stored image file **{page 6,**

paragraph 21; page 18, paragraph 69; page 19, paragraph 75; page 20, paragraph 76; page 21, paragraph 82; Fig. 9, 910};

creating a new directory area for the partition using the directory map **{page 16, paragraph 64; page 18, paragraphs 70-72; page 20, paragraph 76; page 21, paragraph 83; Fig. 7, 708}**; and

protecting the locally-stored image file from accidental user deletion or modification subsequent to creation of the locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file **{page 6, paragraphs 18 and 22; page 11, paragraph 45; page 15, paragraph 60; page 16, paragraph 62; page 20, paragraph 79; page 21, paragraph 83; Fig. 9, 906}**.

21. (Rejected) An apparatus for backing up a file system in a partition comprising a plurality of allocation units, the apparatus comprising:

a processor **{page 6, paragraph 18; page 9, paragraphs 34-36; page 12, paragraph 46; page 15, paragraph 60; page 16, paragraph 62; page 17, paragraph 68; Fig. 5, 504; Fig. 6, 602, 604}**;

a local imager **{page 5, paragraph 17; page 11, paragraphs 42-45; page 12, paragraph 49; page 13, paragraphs 50-52; page 14, paragraphs 54-56;**

page 15, paragraphs 57-59; page 16, paragraph 64; page 20, paragraphs 77-78; Fig. 2, 202; Fig. 3, 202; Fig. 4, 202; Fig. 8, 202} programmed to create a locally-stored image file by copying each allocation unit occupied by a plurality of files of the file system to the locally-stored image file **{page 5, paragraph 17; page 6, paragraph 18; page 11, paragraph 42; page 12, paragraphs 46 and 49; page 20, paragraph 77; Fig. 3, 204, 302, 306; Fig. 9, 902}**, wherein the locally-stored image file is located within the same partition as the file system being backed up **{page 5, paragraph 17; page 11, paragraphs 42, 44, and 45; page 12, paragraph 49; page 20, paragraph 77; Fig. 2, 102, 104, 204; Fig. 9, 902}**, and wherein the local imager is configured to add a directory map to the locally-stored image file that associates copied allocation units in the locally-stored image file with names of corresponding files from the file system **{page 5, paragraph 17; page 14, paragraph 54; page 18, paragraph 70; page 19, paragraph 78; page 21, paragraph 83; Fig. 3, 312; Fig. 9, 904}**; and

a protection component **{page 6, paragraph 18; page 20, paragraph 79; page 21, paragraph 83}** programmed to protect the locally-stored image file from accidental user deletion or modification subsequent to creation of the locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image

file **{page 6, paragraphs 18 and 22; page 11, paragraph 45; page 15, paragraph 60; page 16, paragraph 62; page 20, paragraph 79; page 21, paragraph 83; Fig. 9, 906}**.

31. (Rejected) An apparatus for restoring a file system to a partition comprising a plurality of allocation units, the apparatus comprising:

a processor **{page 6, paragraph 18; page 9, paragraphs 34-36; page 12, paragraph 46; page 15, paragraph 60; page 16, paragraph 62; page 17, paragraph 68; Fig. 5, 504; Fig. 6, 602, 604}**;

an image locater **{page 6, paragraph 20; page 16, paragraph 64; page 17, paragraphs 65-67; page 21, paragraph 81; Fig. 7, 702}** to find a locally-stored image file located within the partition to which the file system is to be restored **{page 6, paragraph 20; page 17, paragraph 66; page 21, paragraph 81; Fig. 9, 908}**, the locally-stored image file comprising a directory map and file data for a plurality of files **{page 5, paragraph 17; page 14, paragraph 54; page 18, paragraph 70; page 19, paragraph 78; page 21, paragraph 83; Fig. 3, 312; Fig. 9, 904}**;

a media formatter **{page 6, paragraph 21; page 16, paragraph 64; page 17, paragraph 68; page 21, paragraph 82; Fig. 7, 704}** to initialize at least a

subset of the allocation units of the partition not occupied by the locally-stored image file including one or more allocation units used for a directory area of the partition **{page 6, paragraph 21; page 17, paragraph 68; Fig. 9, 910}**;

a data extractor **{page 6, paragraph 21; page 16, paragraph 64; page 18, paragraph 69-70; page 21, paragraph 82; Fig. 7, 706}** to extract the file data from the locally-stored image file into the initialized allocation units without disturbing the locally-stored image file **{page 6, paragraph 21; page 18, paragraph 69; page 19, paragraph 75; page 20, paragraph 76; page 21, paragraph 82; Fig. 9, 910}**;

a directory area builder **{page 6, paragraph 22; page 16, paragraph 64; page 18, paragraph 70-72; page 21, paragraph 83; Fig. 7, 708}** to build a new directory area for the partition using the directory map **{page 16, paragraph 64; page 18, paragraphs 70-72; page 20, paragraph 76; page 21, paragraph 83; Fig. 7, 708}**; and

a protection component **{page 6, paragraph 18; page 20, paragraph 79; page 21, paragraph 83}** programmed to protect the locally-stored image file from accidental user deletion or modification subsequent to creation of the locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image

file **{page 6, paragraphs 18 and 22; page 11, paragraph 45; page 15, paragraph 60; page 16, paragraph 62; page 20, paragraph 79; page 21, paragraph 83; Fig. 9, 906}**.

41. (Rejected) A method for localized backup and restoration of a file system in a partition comprising a plurality of allocation units, the method comprising:

creating a locally-stored image file by copying each allocation unit occupied by a plurality of files of the file system to the locally-stored image file **{page 5, paragraph 17; page 6, paragraph 18; page 11, paragraph 42; page 12, paragraphs 46 and 49; page 20, paragraph 77; Fig. 3, 204, 302, 306; Fig. 9, 902}**, wherein the locally-stored image file is located within the same partition as the file system being backed up **{page 5, paragraph 17; page 11, paragraphs 42, 44, and 45; page 12, paragraph 49; page 20, paragraph 77; Fig. 2, 102, 104, 204; Fig. 9, 902}**;

adding a directory map to the locally-stored image file that associates copied allocation units in the locally-stored image file with names of corresponding files from the file system **{page 5, paragraph 17; page 14, paragraph 54; page 18,**

paragraph 70; page 19, paragraph 78; page 21, paragraph 83; Fig. 3, 312; Fig. 9, 904};

locating the locally-stored image file within the partition **{page 11, paragraph 45; page 14, paragraph 56; Fig. 9, 908}**;

initializing at least a subset of the allocation units of the partition not occupied by the locally-stored image file including one or more allocation units used for a directory area of the partition **{page 6, paragraph 21; page 17, paragraph 68; Fig. 9, 910}**;

extracting file data from the locally-stored image file into the initialized allocation units without disturbing the locally-stored image file **{page 6, paragraph 21; page 18, paragraph 69; page 19, paragraph 75; page 20, paragraph 76; page 21, paragraph 82; Fig. 9, 910}**;

creating a new directory area for the partition using the directory map **{page 16, paragraph 64; page 18, paragraphs 70-72; page 20, paragraph 76; page 21, paragraph 83; Fig. 7, 708}**; and

subsequent to creating the locally-stored image file, protecting the locally-stored image file from accidental user deletion or modification by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file. **{page 6,**

paragraphs 18 and 22; page 11, paragraph 45; page 15, paragraph 60; page 16, paragraph 62; page 20, paragraph 79; page 21, paragraph 83; Fig. 9, 906}

42. (Rejected) A computer-readable storage medium comprising program code for backing up a file system in a partition comprising a plurality of allocation units, the computer-readable storage medium comprising:

program code for creating a locally-stored image file by copying each allocation unit occupied by a plurality of files of the file system to a locally-stored image file **{page 5, paragraph 17; page 6, paragraph 18; page 11, paragraph 42; page 12, paragraphs 46 and 49; page 20, paragraph 77; Fig. 3, 204, 302, 306; Fig. 9, 902}**, wherein the locally-stored image file is located within the same partition as the file system being backed up **{page 5, paragraph 17; page 11, paragraphs 42, 44, and 45; page 12, paragraph 49; page 20, paragraph 77; Fig. 2, 102, 104, 204; Fig. 9, 902}**;

program code for adding a directory map to the locally-stored image file that associates copied allocation units in the locally-stored image file with names of corresponding files from the file system **{page 5, paragraph 17; page 14, paragraph 54; page 18, paragraph 70; page 19, paragraph 78; page 21, paragraph 83; Fig. 3, 312; Fig. 9, 904}**; and

program code for protecting the locally-stored image file from accidental user deletion or modification subsequent to creation of the locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file **{page 6, paragraphs 18 and 22; page 11, paragraph 45; page 15, paragraph 60; page 16, paragraph 62; page 20, paragraph 79; page 21, paragraph 83; Fig. 9, 906}**.

43. (Rejected) A computer-readable storage medium comprising program code for restoring a file system to a partition comprising a plurality of allocation units, the computer-readable storage medium comprising:

program code to access a locally-stored image file located within the partition to which the file system is to be restored **{page 6, paragraph 20; page 17, paragraph 66; page 21, paragraph 81; Fig. 9, 908}**, the locally-stored image file comprising a directory map and file data for a plurality of files **{page 5, paragraph 17; page 14, paragraph 54; page 18, paragraph 70; page 19, paragraph 78; page 21, paragraph 83; Fig. 3, 312; Fig. 9, 904}**;

program code to initialize at least a subset of the allocation units of the partition not occupied by the locally-stored image file including one or more

allocation units used for a directory area of the partition **{page 6, paragraph 21; page 17, paragraph 68; Fig. 9, 910}**;

program code to extract the file data from the locally-stored image file into the initialized allocation units without disturbing the locally-stored image file **{page 6, paragraph 21; page 18, paragraph 69; page 19, paragraph 75; page 20, paragraph 76; page 21, paragraph 82; Fig. 9, 910}**;

program code to create a new directory area for the partition using the directory map **{page 16, paragraph 64; page 18, paragraphs 70-72; page 20, paragraph 76; page 21, paragraph 83; Fig. 7, 708}**; and

program code to protect the locally-stored image file from accidental user deletion or modification subsequent to creation of the locally-stored image file by initiating a process at system startup that opens the locally-stored image file to block subsequent processes from accessing the locally-stored image file **{page 6, paragraphs 18 and 22; page 11, paragraph 45; page 15, paragraph 60; page 16, paragraph 62; page 20, paragraph 79; page 21, paragraph 83; Fig. 9, 906}**.

C. Evidence Section

No evidence pursuant to 37 C.F.R. §§ 1.130, 1.131, or 1.132 or entered by the Examiner is being submitted.

D. Related Proceedings Section

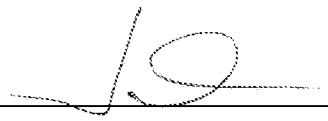
As detailed above, a notice of Appeal and a request for Pre-Appeal Conference were previously filed in this case on 12 March 2008. The Notice of Panel Decision from Pre-Appeal Brief review was mailed on 24 April 2008. The Panel reopened prosecution. Appellant is not aware of any other appeals, interferences, or judicial proceedings that will directly affect, be directly affected by, or have a bearing on the Board's decision in this appeal.

XI. CONCLUSION

For at least the foregoing reasons, Appellant believes that each of the finally rejected claims in this application is in immediate condition for allowance. Accordingly, Appellant respectfully requests the reversal of the rejections of these claims and allowance of the same.

Respectfully submitted,

Date: 24 May 2010



Jonathan R. Lee
Registration No. 56,561